논문 2008-45TC-8-15

# 실시간 네트워크 트래픽 매니지먼트 시스템 구현

## ( Implementation of a Real-time Network Traffic Management System )

용 기 택*, 이 채 우**

( Ki-Tak Yong and Chae-Woo Lee )

요 약

본 논문에서는 ntop, ethereal, netperf, nettest 등 기존의 네트워크 모니터링 시스템의 취약점을 보안하고 동적인 firewall 기능을 가진 네트워크 트래픽 매니지먼트 시스템인 MAGI를 설계, 구현하고 이를 분석한다. MAGI 는 소프트웨어로 개발된 기존의 네트워크 모니터링 시스템과는 달리 소프트웨어와 하드웨어가 결합된 형태를 가진다. 소프트웨어 측면에서는 기존 모니터링 시스템과 차별화된 웹 유저 인터페이스(web user interface) 시스템을 적용하여 트래픽 분석 및 방화벽 적용이 간편할 뿐만 아니라, 하드웨어의 상태를 모니터링 및 관리 할 수 있는 기능을 지닌다. 웹 유저 인터페이스는 PHP 와 MySQL 을 연동하여 데이터베이스(database)를 이용할 수 있어 데이터 분석이 간편한 장점을 지닌다. 하드웨어 파트인 어플라이언스(appliance)는 네트워크에 브릿지(bridge) 모드로 설치하여 모니터링 하는 시스템에 부하가 걸리지 않게 구현하였다. 또한, 어플라이언스 관리를 위하여 ncurses 기반의 인터페이스를 만들어 탑재함으로써 다양한 환경에 적용이 가능하다.

Abstract

In this paper, we will design and substantiate the Network Traffic management system(MAGI), which makes up for the weak points of existing Network Monitoring tools and realize dynamic firewall; MRTG, ntop, ethereal, and nettest. The Network Traffic Monitoring and analyzing system differs from existing software-based Network Monitoring tools as MAGI is a combination of software and hardware. Not only the innovative web user interface applied to the software makes analyzing traffic simpler, but it also has a function for monitoring and managing conditions of the hardware. The web user interface implemented with PHP and MySQL helps to use the database and as a result, analyzing data becomes easier. On the hardware part, the appliance is configured as a bridge in a network. As a result, managed system does not have overload.

**Keywords**: traffic, firewall, monitoring, management

## I. 서 론

최근 인터넷의 발전으로 인해 인터넷은 우리 생활에 매우 많은 부분을 차지하고 있다. 그에 따라온 라인 시스템과 네트워크의 안정적인 운영이 큰 이슈로 떠오르고 있다. 온라인 시스템과 네트워크의안정성에 가장 큰 영향을 미치는 것은 과도한 트래픽 증가이다. 갑작스런 트래픽의 증가가 일정시간 지속되면 네트워크 장비의 결함을 불러올 수 있다. 이러한 네트워크 장비의 결함은 전체 네트워크의 성능과 운영에 매우 큰 영향을 미칠 수 있다. 따라서 네트워크의 트래픽을 실시간으로 분석 하거나 유해 트래픽을 차단 할 수 있는 시스템은 네트워크 트래픽의 양이 점차 증가될 유비쿼터스 시대에 매우 유용하다.

보편적으로 사용하는 네트워크 툴로는 MRTG, SNMP, ntop 등이 있다. MRTG 와 SNMP를 이용한 트래픽 모니터링은 네트워크 전체의 트래픽 변화나 시간 변화에 따른 트래픽 분석에는 효과적이다. 하지만 관리자가 필요로 하는 특정 호스트 검색이나 전체 트래픽 중에서 특정 서비스 및 애플리케이션에 해당하는 트래

* 정회원, 아주대학교 전자공학과
(Ajou University)

※ "본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음" IITA-2008-C1090-0801-0014(2008년도 사업)
접수일자: 2008년4월11일, 수정완료일: 2008년8월12일

픽의 분류나 분석 같은 세부적인 정보 제공은 불가능하다. 또한 기존의 네트워크 모니터링 툴은 트래픽 양이나 프로토콜 종류 등에 대한 분석을 한 뒤 그 결과를 각종 그래프 등으로 보여주기는 하지만, 모니터링 결과를 이용하여 네트워크에 흐르는 트래픽의 차단이나 불안정한 동작을 보이는 네트워크와의 격리 같은 능동적인 조치는 하지 않는다. 따라서 본 논문에서는 기존 네트워크 트래픽 모니터링 툴의 문제점들을 보완하고 방화벽의 기능을 가진 통합적인 네트워크 트래픽 매니지먼트 시스템을 제시한다. 구현된 시스템은 네트워크의 상태를 실시간으로 모니터링 하고 그 결과를 이용하여 방화벽 규칙을 세우거나 변경하는 등 즉각적인 대응을 할 수 있으며, 웹 인터페이스를 통하여 트래픽 분석 결과 조회 및 방화벽 구성이 가능하다.

본 시스템은 다음과 같이 크게 두 가지 형태로 나눌 수 있다. 첫째, 모니터링 결과를 호스트에서 웹으로 조회 할 수 있도록 도와주는 소프트웨어 형태의 '호스트 기반 모니터링 시스템', 둘째, 네트워크 상단에 하드웨어형태로 설치하는 '모니터링&방화벽 시스템' 이다. 하드웨어는 네트워크에 브릿지 형태로 설치되며, 분석된 내역은 데이터베이스로 저장되어 장소와 호스트 컴퓨터에 상관없이 웹에 접속함으로써 언제 어디서든 관리자가 원하는 결과를 조회할 수 있다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 기존의 네트워크 트래픽 모니터링 툴 및 방화벽에 대해 소개하고, Ⅲ장에서는 구현한 네트워크 트래픽 매니지먼트 시스템에 대해 구체적으로 살펴본다. 그리고 Ⅳ장에서는 구현한 네트워크 트래픽 매니지먼트 시스템의 성능 및 개선사항에 대해 알아보고, Ⅴ장에서 결론을 맺는다.

## Ⅱ. 관련 연구

### 1. 모니터링 툴

MRTG(Multi Router Traffic Grapher)[1]는 표준 SNMP 프로토콜을 기반으로 네트워크 트래픽 모니터링과 관리를 위한 공개 소프트웨어이다. 사용자가 지정한 시간단위로 트래픽을 모니터링 하고 결과값을 웹 페이지로 출력하여 준다. 매우 유용한 트래픽 모니터링 툴 중에 하나로 현재 트래픽 모니터링 및 트래픽 관리를 위하여 가장 광범위 하게 쓰이고 있다. MRTG는 원래 라우터의 트래픽 부하량을 측정하기 위하여 개발되었지만 SNMP와 연동하여 네트워크 장비나 시스템의 자원까지 모니터링 할 수 있으며 수집한 데이터를 HTML 문서와 그래픽 파일로 변환해 출력할 수 있다. MRTG는 매우 기본적이면서 강력한 도구임에는 틀림없으나 몇 가지 단점을 가지고 있다. 트래픽 분류 기능이라 상세 분석 기능이 미약하고, 특히 정상 트래픽과 비정상 트래픽의 분류나 프로토콜 및 서비스별 분류 기능이 제공되지 않는다. SNMP(Simple Network Management Protocol)는 네트워크 장비를 감시하기 위한 목적으로 TCP/IP상에 정의된 응용 계층 프로토콜이다.

Ntop[2]은 대표적인 오픈 소스 네트워크 트래픽 모니터링 툴로써 실시간으로 현재 네트워크에 대한 전체적인 상황을 보여주는 프로그램이다. 이는 플로우 기반 네트워크 모니터링 기법을 사용하므로 네트워크 상황에 대한 상세한 정보를 제공한다. Ntop은 매 시간 별로 packet header 정보를 데이터베이스에 저장하여 웹 인터페이스로 분석 결과를 출력할 수 있다. 그러나 하루 이상 장기간의 데이터베이스 저장 및 분석은 지원하지 않으므로 하루 이상의 장기간 네트워크 분석에는 적합하지 않다.

Ethereal[3]은 간단한 설치와 실행으로 편리하게 네트워크에 흐르는 패킷을 캡쳐하여 트래픽 정보를 모니터링 할 수 있는 프로그램이다. Ethereal은 수집된 트래픽 데이터를 저장하거나 다시 꺼내볼 수 있으며 인터페이스(interface) 환경은 콘솔 버전과 그래픽 버전이 있다. Ethereal을 실행하게 되면 Ethereal이 설치된 컴퓨터의 NIC(Network Interface Card)를 통과하는 모든 트래픽을 볼 수 있으며 Ethereal은 트래픽을 순서대로 나열하고, Source Address, Destination Address, Protocol type, Packet Information등의 정보를 보여준다. Ethereal 은 수집한 패킷 정보를 다양한 통계 결과로 보여주며 다양한 기준으로 필터링도 가능하다. Ethereal은 매우 편리하고 다양한 기능을 가진 강력한 툴이지만 데이터 베이스 관련 기능들을 지원하지 않아 장기간의 트래픽 모니터링 및 분석에는 적합하지 않다.

기존의 네트워크 트래픽 모니터링 툴은 데이터 베이스 저장 및 관리의 기능이 취약하고, 단순한 트래픽 모니터링 기법으로 분석 결과가 다양하지 못하다. 또한 소프트웨어 형태로 제공되므로 관리하고자 하는 네트워크에 대한 능동적인 대처를 할 수 없다. 본 논문에서는 이러한 문제점과 한계들을 개선 및 보완하여 장기간 트래픽 모니터링 및 분석에 용이하며, 네트워크의 상태에 따라 능동적인 대처를 할 수 있는 네트워크 모니터링 시스템을 개발하고자 한다.

### 2. 방화벽(Firewall)

방화벽이란 네트워크를 신뢰 할 수 있는 네트워크와 신뢰 할 수 없는 네트워크로 나누어 그 사이에 흐르는 정보의 흐름을 차단하거나 허용하는 소프트웨어와 하드웨어 모두를 포괄하는 개념이다. 방화벽의 동작방식은 패킷 필터링(Packet filtering) 방식, 어플리케이션 게이트웨이(application gateway) 방식, 상태추적(Stateful Inspection) 방식, 동적 필터링(Dynamic packet filtering) 방식, 하이브리드(hybrid) 방식 등으로 약 5가지로 분류 할 수 있다.

패킷 필터링 방식[4~5]은 OSI 7 layer 중에서 Network Layer와 Transport Layer에서 동작하며 패킷 헤더의 Source IP와 Destination IP 그리고 Source port와 Destination port의 정보를 가지고 패킷을 차단하거나 허용한다. 장점으로는 어플리케이션 게이트웨이 방식에 비해 처리 속도가 빠르고 운용방법이 편리하다. 하지만 data의 대한 직접적인 해석이 불가능하여 바이러스에 감염 된 파일을 가지고 있는 메일이나 파일의 전송을 막지 못한다. 또한 방화벽의 규칙의 수와 순서에 따라 부하가 가중된다.

어플리케이션 게이트웨이 방식[4~5]은 패킷의 실질적인 데이터를 기초로 패킷을 차단하거나 허용한다. 이 방식은 각 서비스 별로 프로시 데몬이 실행되어 클라이언트와 서버 사이의 접속을 관리한다. 내부 시스템에 접속하기 위해서는 방화벽의 프록시를 통해서만 접속이 허용 되고 내부 네트워크와의 직접적인 연결은 허용되지 않는다. 따라서 방화벽 안의 내부 네트워크의 IP 주소를 숨길수 있다. 이 방식의 장점은 프록시의 사용으로 보안성이 패킷 필터링 보다 우수하며 네트워크의 부하가 줄어든다. 하지만 서비스 별로 데몬이 필요하므로 시스템 구성의 복잡도가 커지고 메신져나 P2P 프로그램 같은 프로그램에서 사용할 수 없다는 단점이 있다.

상태 추적 방식[4~5]에서는 SYN패킷에 의해 생성된 세션 테이블을 이용하여 고속으로 패킷을 처리한다. 즉 테이블에 기록 되어진 패킷들과 관련된 다음 패킷은 보안절차 없이 바로 고속으로 패킷을 처리한다. 패킷 필터링과 어플리케이션 게이트웨이 방식의 단점을 보완한 기술로써 패킷의 상태 정보를 이용하여 빠르고 높은 보안성을 제공한다.

동적 필터링 방식[4~5]은 용어 그대로 실제 접속 상태를 반영하여 동적으로 방화벽의 허용 조건이 바뀌는 방식이다. IP 주소와 포트 번호등과 같은 세션 정보를 기록하여 능동적인 관리가 가능하다.

하이브리드 방식[4~5]은 대부분의 방화벽에서 사용되는 방식이다. 패킷 필터링과 어플리케이션 게이트웨이 방식을 혼합하여 사용하는 방식으로 사용자와 네트워크 환경에 유연하게 방화벽을 구성할 수 있지만 두 가지 방식이 결합되어 관리상 복잡하다는 단점이 있다.

본 논문에서는 트래픽 모니터링 시스템을 이용하여 네트워크 트래픽을 분석하고 그 결과를 이용하여 방화벽을 구성하고 규칙을 변경할 수 있는 방화벽 시스템을 구현하고 네트워크 트래픽 모니터링 시스템과 방화벽 시스템을 결합하여 네트워크 트래픽 매니지먼트 시스탬을 개발하고자 한다.

## III. 구현된 네트워크 트래픽 매니지먼트 시스템

제안하는 네트워크 트래픽 매니지먼트 시스템(이하 MAGI로 칭함)의 구체적인 설계 목표 성능 및 기능은 다음과 같다. 첫째, 분석 시스템은 하드웨어 장비(어플라이언스)와 소프트웨어 장비로 나누어 개발한다. 따라서 모니터링 하고자 하는 대상 시스템이나 대상 네트워크의 리소스를 사용하지 않고 독립된 하드웨어로 동작할 수 있다. 둘째, 분석결과를 데이터베이스에 저장한다. 즉 실시간으로 패킷을 분석할 수 있을 뿐만 아니라, 데이터베이스에 저장된 과거의 데이터까지 분석할 수 있다. 셋째, 대부분의 네트워크 환경에서 사용할 수 있도록 브릿지 형태로 제작하며, 목표 성능은 100Mbps급 이다. 넷째, 방화벽은 동적 필터링 방식으로 설계 되어 네트워크의 능동적인 관리를 할 수 있다.

MAGI 는 그림 1과 같이 구성된다. MAGI는 브릿지로서 동작하는데, 상위 WAN포트는 라우터와 연결되어

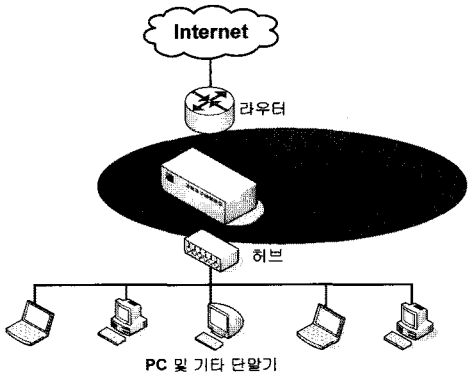


그림 1. 전체 시스템 구성도
Fig. 1. The entire network organization.

표 1. 하드웨어 구성표
Table 1. Hardware line up.

| | | |
|---|---|---|
| 하드웨어 | CPU | AMD athlon 1Ghz * 1 |
| | Memory | 512 Mbyte * 1 |
| | HDD | 5400 rpm * 1 |
| | NIC | 100Mbps * 3 |

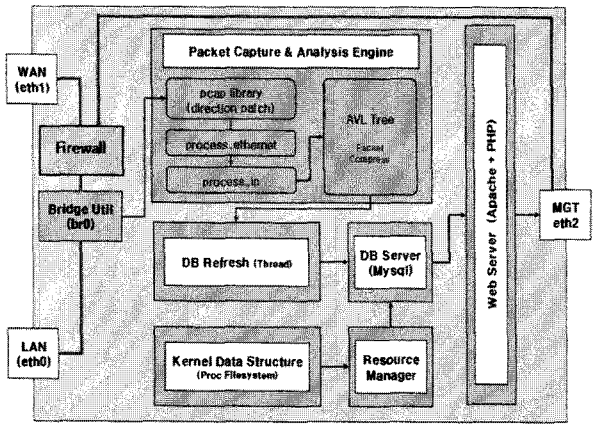


그림 2. MAGI 내부 구조
Fig. 2. Internal process model of MAGI.

있고 하위 LAN포트는 L2 스위치(더미 / 스위칭 허브)와 연결되어 있다. 그림1에서는 라우터와 허브 사이에 위치하면서 하위 단에 연결된 소규모 네트워크로 통하는 모든 트래픽을 수집 / 분석 하게 된다. 분석된 결과를 토대로 하드웨어를 통과하는 패킷을 차단하거나 허용하게 된다. 또한 MAGI는 브릿지 모드로 동작하기 때문에 매우 유연하게 네트워크에서 관리자가 원하는 곳에 연결할 수 있다. 그러므로 연결하는 위치에 따라 하나의 단말기에 통하는 트래픽 측정부터 좀 더 광범위한 네트워크의 트래픽 측정 까지 매우 다양하게 적용할 수 있다.

MAGI프로젝트 진행에 사용된 하드웨어의 사양은 표1과 같다. 브릿지 모드로 동작하기 위하여 2개의 NIC를 사용하였고 나머지 1개의 NIC는 분석된 트래픽 정보를 열람하거나 관리하기 위한 용도로 사용되었다. HDD는 모니터링 된 트래픽 정보를 저장하기 위하여 사용되었다.

### 1. 동작구조

MAGI의 내부 동작은 브릿지 모드로 네트워크에 연결하고, eth1과 eth0 두 개의 NIC 사이에 흐르는 패킷을 캡쳐하면서 시작된다. 캡쳐된 패킷은 Packet Capture & Analysis Engine으로 보내진다. Packet Capture & Analysis Engine에서 분석된 패킷들의 데이

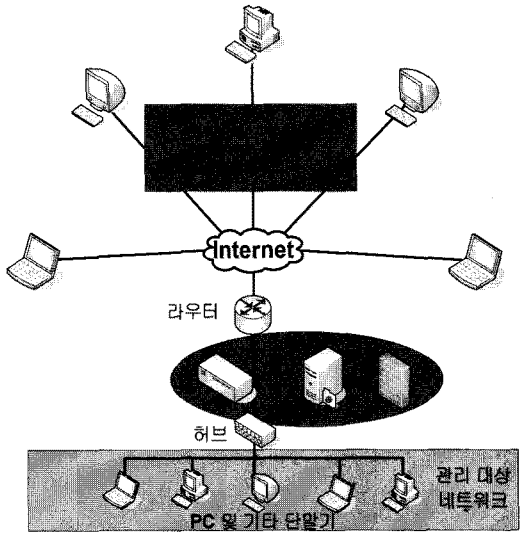


그림 3. 네트워크 트래픽 매니지먼트를 위한 네트워크 구성의 예
Fig. 3. Example of network organization for network traffic management.

터는 DB Refresh 모듈로 보내져 압축한 뒤 데이터 베이스에 기록된다. 이렇게 기록된 데이터는 Resource Manager 모듈에서 보내온 시스템 데이터와 함께 Web Server 모듈로 전송되고, Web Server모듈은 eth2 NIC를 이용하여 실시간 트래픽 현황 및 시스템의 자원을 웹 상으로 열람할 수 있게 한다. 모니터링 된 결과를 토대로 웹에서 방화벽을 규칙을 세우거나 변경하면 eth2로 방화벽 관련 명령어들이 전송 되어 진다. eth2가 받은 명령어는 firewall 모듈에서 방화벽을 관리한다.

그림 2에서 MAGI의 내부 동작 방식을 확인할 수 있다. MAGI의 외부 동작 방식은 그림 3에서 확인할 수 있다. 라우터와 관리 대상 네트워크 사이에 설치된 MAGI의 하드웨어 시스템은 관리 대상 네트워크로 흐르는 모든 패킷을 수집 및 분석한다. 모니터링 한 데이터는 인터넷 액세스가 가능한 컴퓨터라면 언제 어디서나 시간과 장소에 구애 받지 않고 분석된 패킷 데이터의 조회 및 하드웨어의 관리가 가능하다. MAGI의 분석 엔진은 최근 각광 받는 플로우 기반 분석 방식을 사용하고 있어 관리 대상 네트워크의 세밀한 분석이 가능하다.

### 2. 내부 모듈

각각의 모듈 설명은 다음과 같다.

· Bridge environment : 두 개의 NIC를 브릿지로 구성하여 네트워크에 inline으로 설치한다. 리눅스 커널에서 제공하는 bridging[4] 옵션을 사용하여 컴파일 하

고, 제어는 brctl이라는 공개된 유틸리티를 이용하였다.

· Packet Capture & Analysis Engine : 패킷의 수집은 pcap library[5]를 이용하였다. 단, pcap만으로는 inbound / outbound 와 같은 트래픽의 direction을 알 수 없으므로, 트래픽의 direction[6] 정보를 알 수 있게 해주는 patch를 적용하였다. 패킷이 처리되는 과정을 설명하면 다음과 같다. Pcap라이브러리로 수집한 데이터는 processing_packet함수로 전달된다. 수집된 패킷은 process_avl_ethnet (Ethernet frame), process_avl_ip (ip datagram) 함수를 통하여 분석되며, 분석된 결과는 AVL Tree에 모인다. AVL Tree에 있는 데이터를 처리하기 위하여, 정해진 시간마다 한번씩 쓰레드(refresh_ db)가 동작한다. 쓰레드는 AVL Tree에 있는 데이터를 각각 데이터베이스 안에 있는 Ethernet / ip 관련 테이블에 write한다. 쓰레드가 데이터를 데이터베이스에 write하는 동안에도 패킷은 정상적으로 계속 AVL Tree에 모인다.

· Analysis algorithms : 패킷은 locality 가 높은 데이터 형태이다. 즉, 헤더 정보가 다른 패킷이 균일하게 발생되는 것이 아니라 헤더 정보가 일치하는 데이터가 순간적으로 들어오는 경우가 많다. 이러한 패킷의 성질을 이용하면 처리해야 할 패킷의 수를 크게 줄일 수 있다. 이 때 패킷 분석의 빠른 처리 속도를 위하여 AVL Tree 알고리즘을 선택하였다. 데이터베이스의 효율적인 데이터 관리를 위해서는 search와 insert연산에 대해 좋은 효율을 보여주어야 하는데, AVL Tree는 두 가지의 연산에 대하여 최악의 경우에도 일정한 시간복잡도를 보장한다. 시간복잡도란 자료구조에서 알고리즘이 시작하여 답을 얻어 알고리즘이 끝날 때까지의 계산 시간을 의미한다.

· DB Refresh : 일정 시간 동안 패킷을 AVL Tree에 모으면서 압축한 뒤, 데이터베이스에 기록한다. 패킷이 데이터베이스에 기록되는 동안에도 지연 없이 정상적으로 처리되어야 한다. 따라서 새로운 쓰레드(Thread)를 생성한 뒤, 그림 2에서 볼 수 있는 DB Refresh 모듈에서 병렬적으로 데이터베이스 연산을 수행하도록 하였다.

· Resource Manager : CPU, Memory, Disk의 사용률이나, 네트워크의 실시간 in/out bound traffic 양과 같은 시스템 자원을 모니터링 하는 데몬이다. Resource Manager 모듈에서는 커널의 자료구조에 접근하기 위하여 proc file system을 이용한다.

· DB Server : 패킷 분석은 대부분 데이터베이스 연산을 통하여 계산된다. 따라서 가벼우면서도 강력한 데이터베이스가 필요하다. MySQL[7]은 이 조건을 만족시킬 뿐만 아니라 무료로 사용할 수 있고, 많은 분야에서 사용된다. 또한 다양한 드라이버와 어플리케이션이 존재해 다른 응용프로그램과 연동하기에도 편리하다.

· Web Server : 모니터링한 데이터를 웹 유저 인터페이스환경으로 편리하게 분석하기 위하여 Web Server를 구성하였다. Web Server는 apache + php로 구성되었다.

· Firewall : Firewall 모듈은 웹에서 전송한 명령어를 eth2로 받아 방화벽을 구성하는 모듈이다. 방화벽은 리눅스 커널에서 제공하는 iptable[8]을 이용하여 구축하였다. iptable은 기본적으로 패킷 필터링 방식으로 동작한다. 하지만 MAGI에서는 사용자가 실시간으로 네트워크를 모니터링하고 즉각적으로 네트워크 변화에 반응하여 방화벽을 구성하므로 Firewall 모듈은 동적 필터링 방식으로 동작한다고 할 수 있다. 그림 4[9]를 보면 Firewall 모듈은 크게 세 가지 체인으로 동작한다. 첫째 외부에서 자신으로 들어오는 패킷을 처리하는 입력 체인, 둘째 외부에서 자신 이외의 목적지로 들어오는 패킷을 처리하는 전달 체인, 셋째 내부에서 외부로 나가는 패킷을 처리하는 출력 체인이다. 세 가지 경우 모두 패킷이 각각의 체인 규칙에 부합하면 패킷을 정해진 프로세스에 넘겨주고 반대의 경우에는 패킷을 드롭한다.

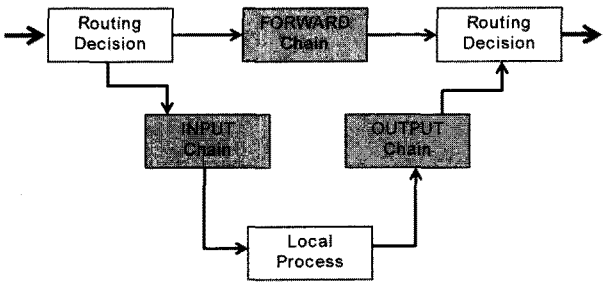


그림 4. Firewall 프로세싱 구조
Fig. 4. Process structure of the MAGI Firewall.

## 3. 주요 기능

본 절에서는 MAGI의 주요 기능을 나열하고 각각의 기능에 대해서 설명한다.

· 시스템 정보 : 시스템정보 기능은 MAGI 하드웨어의 전체적인 상황을 보여준다. 즉 하드웨어의 CPU사

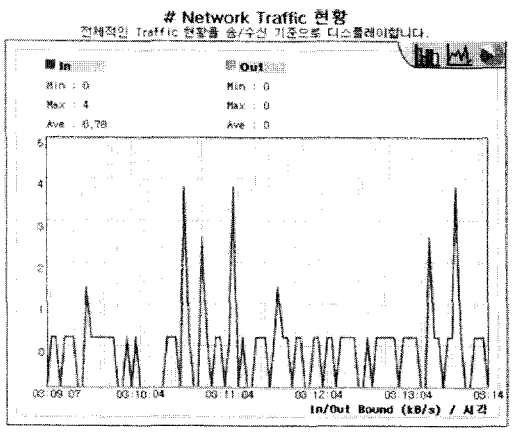


그림 5. 트래픽 분석의 예
Fig. 5. Real-Time traffic load graph in MAGI.

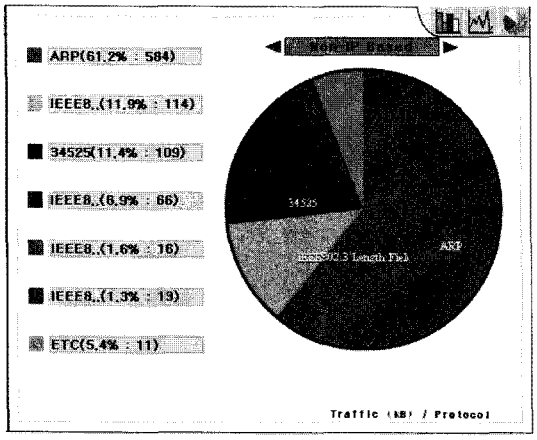


그림 6. 이더넷 프레임 분석의 예
Fig. 6. Analyzed information of an ethernet frame.

| 순위 | Port | Service Name | iFlows | Traffic | | |
|---|---|---|---|---|---|---|
| | | | | Send | Receive | Total |
| 0 | 22 | http | 10 | 0 KB | 0 KB | 0 KB |
| 1 | 4231 | http | 1275 | 0 KB | 0 KB | 0 KB |
| 2 | 137 | http | 1886 | 0 KB | 0 KB | 0 KB |
| 3 | 138 | http | 221 | 0 KB | 0 KB | 0 KB |
| 4 | 1900 | http | 80 | 0 KB | 0 KB | 0 KB |
| 5 | 4232 | http | 30 | 0 KB | 0 KB | 0 KB |

그림 7. 포트 분석의 예
Fig. 7. Analysis of port traffic.

용량, Memory 사용량, Disk 사용량을 실시간 그래프로 표현해준다. 또한 현재 브릿지 모드로 동작하는 하드웨어를 통과한 트래픽 양을 시간축의 그래프로 보여준다. 또한 Alert 창을 통한 하드웨어의 과부하 상태를 미리 알 수 있다. 그림 5는 하드웨어를 통과하는 트래픽의 양을 실시간 그래프로 보여준다.

· 트래픽 분석 : 그림 6은 네트워크 트래픽의 이더넷 프레임 분석결과 이다. 이 파트는 프로토콜의 종류(IP, ARP등), 트래픽의 양, 패킷 개수, 패킷 사이즈별 구성 비율 등의 정보를 보여준다.

· IP 프로토콜 분석 : IP 프로토콜 메뉴는 IP header 정보를 분석한다. IP 내부 프로토콜 종류(TCP, ICMP 등), In/Out 여부, TTL, 트래픽 양, 패킷의 개수, 캐스팅 방식(Uni Casting/Multi Casting/Broad Casting) 등의 정보를 나타낸다.

· 서비스 포트 분석 : TCP/UDP 포트에 대한 정보를 분석한다. 포트별 사용 트래픽 양, 패킷 개수, 포트 사용 IP 등의 정보를 나타낸다. 그림 7에서 확인 할 수 있다.

· ncurses 기반의 하드웨어 관리 interface : 하드웨어의 간단한 설치 및 사용을 위하여 linux OS 의cmd모드가 아닌 키보드의 화살표키와 숫자 키를 이용하여 모든 설정 및 관리를 할 수 있도록 제작 하였다.

· 방화벽 기능 : MAGI를 통과하는 트래픽을 분석하여 관리 대상 네트워크로 흐르는 패킷을 허락하거나 차단 하는 시스템이다. 목적지 주소, 소스 주소, 목적지 포트, 소스 포트 등의 네 가지 항목으로 방화벽 규칙을 설정할 수 있다. 방화벽의 룰은 상위 번호가 더 높은 우선권을 가진다. 그림8에서 확인 할 수 있다.

## Ⅳ. 시스템의 성능 및 개선사항

### 1. 소프트웨어

첫째, 현재 Ethernet이나 IP정보 등에 따라서 각각 다른 테이블에 저장하고 있지만, 트래픽 모니터링 특성상 시간이 지날수록 각각의 테이블에는 많은 양의 데이터가 쌓인다. 본 논문에서는 index를 사용하여 query의 수행속도를 향상 시키고 있지만, 이 경우에는 insert의 속도를 희생할 수 밖에 없는 단점이 있다. 이 단점의 해결 방안은 일정한 시간대 별로 다른 테이블을 이용할 수 있도록 데이터베이스 구조를 수정하는 것이다. 이렇게 하면 시간이 지날수록 테이블에 들어가는 데이터 갯수가 증가하는 현상을 막을 수가 있다.

둘째, MAGI는 브릿지 모드로 동작하기 때문에 iptable[8]을 이용하면 손쉽게 방화벽을 구성할 수 있다. 이 때, MAGI의 트래픽 분석 결과를 이용하여 방화벽 규칙에 사용할 수 있다. 예를 들어 특정 IP가 전체 트래픽의50% 이상을 10분 이상 차지하고 있을 경우 자동으로 방화벽 차단룰에 추가될 수 있도록 설정할 수 있다. 또한 이렇게 네트워크 모니터링 시스템과 방화벽이 하

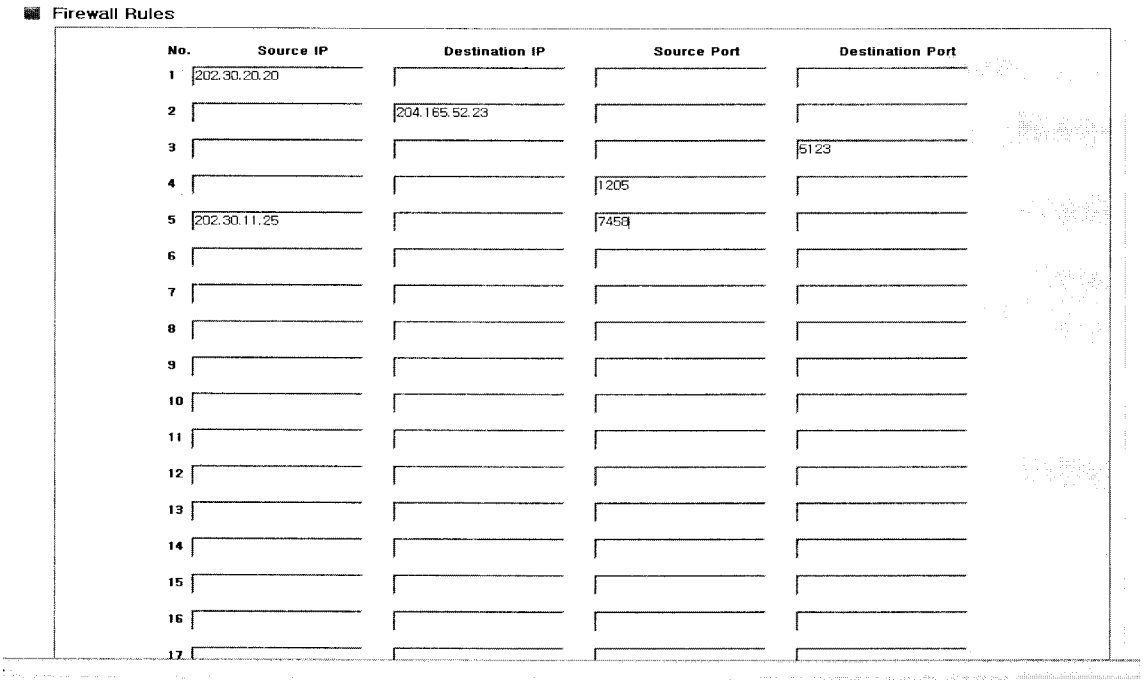


그림 8. 방화벽 규칙 적용의 예
Fig. 8. Example of firewall rules.

나의 어플라이언스에서 동작할 경우, 네트워크 관리자가 방화벽의 규칙을 만들 때 모니터링 결과를 참고할 수 있다.

## 2. 하드웨어

MAGI의 성능을 이론적으로 계산하여 보면 약 140Mbps 이다. 이것은 다음과 같은 방법으로 얻을 수 있다. 이더넷 레벨에서 헤더 길이를 제외한 MTU 사이즈는 약 1500byte 이며 데이터베이스는 레코드 하나당 하나의 패킷 정보를 저장 하므로 약 (1500*10000)/1024/1024 = 14Mbps의 성능을 얻을 수 있다. 여기서 10000은 테스트를 통하여 얻은 데이터베이스의 초당 패킷 처리 양이다. 이것은 트래픽 모니터링 시스템에 들어오는 모든 패킷을 처리하였을 때의 결과이다. 네트워크 트래픽의 특성상 중복된 패킷이 많이 발생한다. 따라서 우리는 AVL Tree를 이용하여 중복 되는 패킷을 하나의 파일로 압축하여 저장해야 할 데이터를 갯수를 평균 10배 정도 줄였다. 따라서 MAGI의 이론적인 성능은 약 140Mbps이다. 하지만 구현된MAGI system의 성능은 약 65Mbps이다. 이 이유는 시스템을 구현할 때 테스트한 시스템과는 달리 저사양의 하드웨어를 사용하였기 때문이다. 최근 트래픽 모니터링 시스템에서 사용되는CPU를 살펴보면 보통 듀얼 코어 CPU를 사용하며,

표 2. 하드디스크 성능 비교 결과
Table 2. Benchmarking result.

| | |
|---|---|
| [illegible] 5400rpm | 66MB in 3.02 sec = 21.87 MB/sec |
| [illegible] 7200rpm | 157MB in 3.01 sec = 52.15 MB/sec |

이론적인 성능을 계산 할 때에 사용했던 시스템 역시 듀얼 코어 시스템이었다. 하지만 MAGI의 시스템은 싱글 코어의 1.1Ghz CPU를 사용한다. 따라서 당초 목표로 했던 100Mbps급의 모니터링 시스템을 만들기 위해서는 현재 MAGI가 가지고 있는 1.1Ghz급 이상의 CPU가 필요하다. 또한 MAGI에 사용중인 하드 디스크의 스펙은 DMA-66, 5400rpm 이다. 메인보드가 지원 가능한 상위 기종의 하드디스크(DMA-100, 7200rpm)를 사용하여 비교 하여 보니 확연한 성능 차이를 알 수 있었다.

MAGI의 하드웨어를 교체하기 위해서는 새로운 하드웨어에 맞춰서 커널 환경을 다시 구성해야 한다. MAGI와 같이 데이터베이스를 많이 사용하는 환경에서는 특히 하드디스크의 성능이 전체 성능에 많은 영향을 미친다. 현재 MAGI의 최대 속도는 약 65Mbps이다. 목표로 한 100Mbps는 표 2의 결과를 고려해 볼 때 하드디스크

의 업그레이드만으로도 우리가 달성하고자 하는 목표에 도달할 수 있을 것으로 예상된다.

## V. 결 론

MAGI는 실시간 네트워크 트래픽 매니지먼트 시스템으로 다양한 환경에 적용할 수 있도록 설계되었다. MAGI는 소프트웨어와 하드웨어로 구성 되어 하드웨어에서는 트래픽 데이터를 저장 하고 저장된 데이터를 분석한다. 소프트웨어에 파트는 웹 유저 인터페이스 기반으로 구현된 트래픽 분석 및 방화벽 툴이다. 이 툴은 분석된 데이터를 여러 가지 모드의 그래프로 나타낼 수 있고 방화벽을 관리할 수 있다. 마지막으로 브릿지 모드로 동작하는 하드웨어는 네트워크상의 어느 곳이나 유연하게 설치 가능하다. 향후 계획은 위에서 언급한 개선사항을 개선하여 더 큰 대역폭에서 동작하는 트래픽 모니터링 및 분석 시스템을 구현 하는 것이다.

## 참 고 문 헌


[1] MRTG, "http://oss.oetiker.ch/mrtg/".
[2] ntop, "http://www.ntop.org".
[3] Ethereal, "http://www.ethereal.com".
[4] Ethereal, "http://www.ethereal.com".
[5] C. Roeckl, "*Stateful Inspection Firewalls*", Juniper Networks, Inc., 2004.
[4] W. R. Stevens, "*Advanced Programming in the UNIX Environment*", Addison-Wesley, 1992.
[5] Pcap library, "http://www.tcpdump.org".
[6] Pcap option, "http://www.beonline.net.au.kb/pcap.html".
[7] MySQL, C API, "http://www.kl.org".
[8] iptable, "http://iptables-tutorial.frozentux.net /iptables-tutorial.html".
[9] Linux Network Administrators Guide, "http://www.faqs.org/docs/linux_network".


## 저 자 소 개


용 기 택(정회원)
2008년 아주대학교 전자공학과 학사 졸업.
2008년 3월 ~ 현재 아주대학교 전자공학과 석사 과정
<주관심분야 : Traffic Engineering, Sensor Netwrok, >

이 채 우(정회원)
1985년 서울대학교 제어계측 학사 졸업.
1988년 한국과학기술원 전자공학과 석사 졸업.
1995년 University of Iowa 박사 졸업.
1985년 1월 ~ 1985년 12월 (주)금성통신 연구원.
1988년 9월 ~ 1999년 3월 한국통신 선임연구원.
1999년 3월 ~ 2001년 9월 Lucent Technologies Korea 이사.
2001년 9월 ~ 2002년 2월 한양대학교 겸임교수
2002년 3월 ~ 현재 아주대학교 전자공학과 부교수.
<주관심분야 : 광대역 통신망, Ubiquitous networking, Traffic Engineering>